

# Binary Constraint Satisfaction Problems Defined by Excluded Topological Minors\*

David A. Cohen  
Royal Holloway, University of London  
dave@cs.rhul.ac.uk

Martin C. Cooper  
IRIT, University of Toulouse  
cooper@irit.fr

Peter G. Jeavons  
University of Oxford  
peter.jeavons@cs.ox.ac.uk

Stanislav Živný  
University of Oxford  
standa.zivny@cs.ox.ac.uk

## Abstract

The binary Constraint Satisfaction Problem (CSP) is to decide whether there exists an assignment to a set of variables which satisfies specified constraints between pairs of variables. A binary CSP instance can be presented as a labelled graph encoding both the forms of the constraints and where they are imposed. We consider subproblems defined by restricting the allowed form of this graph. One type of restriction that has previously been considered is to forbid certain specified substructures (patterns). This captures some tractable classes of the CSP, but does not capture classes defined by language restrictions, or the well-known structural property of acyclicity.

In this paper we extend the notion of pattern and introduce the notion of a topological minor of a binary CSP instance. By forbidding a *finite* set of patterns from occurring as topological minors we obtain a compact mechanism for expressing novel tractable subproblems of the binary CSP, including new generalisations of the class of acyclic instances. Forbidding a finite set of patterns as topological minors also captures all other tractable structural restrictions of the binary CSP. Moreover, we show that several patterns give rise to tractable subproblems if forbidden as topological minors but not if forbidden as sub-patterns. Finally, we introduce the idea of augmented patterns that allows for the identification of more tractable classes, including all language restrictions of the binary CSP.

## 1 Introduction

The Constraint Satisfaction Problem (CSP) is to decide whether it is possible to find an assignment to a set of variables which satisfies constraints between certain subsets of the variables. This paradigm has been applied in diverse application areas such as Artificial Intelligence, Bioinformatics and Operations Research [38, 28].

As the CSP is known to be NP-complete, much theoretical work has been devoted to the identification of tractable subproblems. Important tractable cases have been identified by restricting the hypergraph *structure* of the constrained subsets of variables [26, 34]. Other tractable cases have been identified by restricting the forms of constraints (sometimes called the constraint *language*) [29, 22]. Work on both of these areas is now very far advanced: a full complexity dichotomy for any structural or language restriction now requires the classification of just one remaining family of languages [2, 3]. Moreover, it is known that a full characterisation of the

---

\*An extended abstract of part of this work appeared in the *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI'15)* [10]. The authors were supported by EPSRC grant EP/L021226/1. Stanislav Živný was supported by a Royal Society University Research Fellowship.

complexity of the *binary* CSP (where constraints restrict the assignment to precisely two variables) would enable a full characterisation of the complexity of all language restrictions for the general CSP with constraints of any arity [22].

However, identifying the subproblems obtained by restricting the language or the structure of a CSP alone is not a sufficiently rich framework in which to investigate the full complexity landscape. For example, we may wish to identify all the instances solved by a particular algorithm, such as enforcing arc-consistency [17, 38]. It has been shown [22, 11] that this class of instances includes all instances defined by a certain structural restriction, together with all instances defined by a certain language restriction, as well as further instances that are not defined by either kind of restriction alone. Hence we need a more flexible mechanism for describing subproblems that will allow us to unify and generalise such descriptions.

Here we develop a new mechanism of this kind that uses certain tools from graph theory to define restricted classes of labelled graphs that represent binary CSP instances. Our mechanism allows us to impose simultaneous restrictions on both the structure and the language of an instance, and hence obtain a more refined collection of subproblems, allowing a more detailed complexity analysis. Subproblems of the CSP of this kind are sometimes referred to as *hybrid* subproblems and, currently, very little is known about the complexity of such subproblems [6].

The tools that we use to obtain restricted classes of labelled graphs build on a well-established line of research in graph theory, by considering local “obstructions” or “forbidden patterns”. The idea of using forbidden patterns has previously been applied to the binary CSP and resulted in the discovery of a number of new tractable classes [8, 9, 13, 21]; related ideas also appeared in [33, 31]. However, we will show in this paper that in order to unify structural, language and existing hybrid restrictions within a single framework we need even more flexibility.

In graph theory it proved useful to go beyond the idea of forbidden subgraphs and introduce the more flexible concept of forbidden minors. A well-known result of Robertson and Seymour states that any set of graphs closed under the operation of taking minors is specified by a finite set of forbidden minors. Rather than adapt the full machinery of graph minors to the CSP framework, we consider here the slightly simpler notion of a *topological minor* [18]. We show that by adapting the notion of topological minor to the CSP framework we are able to provide a unified description of all tractable structural classes, all tractable language classes, and some new tractable classes that cannot be captured as either structural classes or language classes. Moreover, we are able to show that the class of tree-structured CSP instances has a very simple description in this framework, and there exist tractable classes of the binary CSP that properly extend this class and yet still have a very simple description.

## 2 Preliminaries

We define the constraint satisfaction problem in Section 2.1 and patterns in Section 2.2. Section 3 then introduces the idea of defining restricted classes of CSP instances by forbidding certain patterns to occur as sub-patterns or topological minors.

### 2.1 The CSP

Constraint satisfaction is a paradigm for describing computational problems. Each problem instance is represented as a constraint network: a collection of variables that take their value from some given domain. Some subsets of the variables have a further restriction on their allowed simultaneous assignments, called a constraint. A solution to such a network assigns a value to each variable such that every constraint is satisfied.

In this paper we consider only *binary* constraint networks, where every constraint limits the values of precisely two variables. It has been shown that any constraint network can be reduced to an equivalent binary network over a different domain of values [16, 37].

**Definition 2.1** An instance of the binary constraint satisfaction problem (CSP) is a triple  $(V, D, C)$  where  $V$  is a finite set of variables, for each  $v \in V$ ,  $D(v)$  is a finite domain of values for  $v$ , and  $C$  is a set of constraints, containing a constraint  $R_{uv}$  for each pair of variables  $(u, v)$ . The constraint  $R_{uv} \subseteq D(u) \times D(v)$  is the set of compatible assignments to the variables  $u$  and  $v$ .

A solution to a binary CSP instance is an assignment  $s : V \rightarrow D$  of values to variables such that, for each constraint  $R_{uv}$ ,  $(s(u), s(v)) \in R_{uv}$ .

We will assume that there is *exactly one* binary constraint between any two variables. That is, if we define  $R'_{uv}$  as  $\{(b, a) \mid (a, b) \in R_{uv}\}$ , then  $R_{vu} = R'_{uv}$ . This is just a notational convenience since we can pre-process each instance, replacing  $R_{uv}$  with  $R_{uv} \cap R'_{vu}$ . A constraint will be called *trivial* if it is equal to the Cartesian product of the domains of its two variables.

We will say that a class of CSP instances is *tractable* if there is a polynomial-time algorithm to decide whether any instance in the class has a solution. The size of a constraint instance will be taken to be the sum of the sizes of the constraint relations. If the size of the domain for any variable and the arity of the constraints is bounded, then this is polynomial in the number of variables.

Note that Definition 2.1 describes a standard form of mathematical specification for a CSP instance that is convenient for theoretical analysis. In the next subsection we will introduce an alternative representation in terms of patterns (see Construction 2.5). Often more concise representations are used, and trivial constraints are usually not represented [38].

Arc consistency (AC) is a fundamental concept for the binary CSP [17, 38].

**Definition 2.2** A pair of variables  $(u, v)$  is said to be *arc-consistent* if for each value  $a \in D(u)$  in the domain of  $u$ , there is a value  $b \in D(v)$  in the domain of  $v$  such that  $(a, b) \in R_{uv}$ .

A binary CSP instance is *arc consistent* if every pair of variables is arc-consistent.

Given an arbitrary CSP instance  $I$  there is a unique smallest set of domain values which can be removed to make the instance arc-consistent. Furthermore the discovery of this unique minimal set of domain values and their removal, called establishing arc-consistency, can be done in polynomial time [12]. For a given instance  $I$  we will denote by  $AC(I)$  the instance obtained after establishing arc-consistency.

## 2.2 Patterns

We now introduce the central notion of a pattern, which can be thought of as a labelled graph, with three distinct kinds of edges.

**Definition 2.3** A pattern is a structure  $(X, E^{\sim}, E^+, E^-)$ , where

- $X$  is a set of points;
- $E^{\sim}$  is a binary equivalence relation over  $X$  whose equivalence classes are called parts;
- $E^+$  is a symmetric binary relation over  $X$  whose tuples are called positive edges;
- $E^-$  is a symmetric binary relation over  $X$  whose tuples are called negative edges.

The sets  $E^{\sim}$  and  $E^+$  are disjoint, and the sets  $E^{\sim}$  and  $E^-$  are disjoint.

In a general pattern there may be pairs of points  $x$  and  $y$  in distinct parts such that  $(x, y)$  is neither a positive nor a negative edge, and there may be pairs of points  $x$  and  $y$  in distinct parts such that  $(x, y)$  is *both* a positive and a negative edge. A pattern is called *complete* if every pair of points  $x$  and  $y$  in distinct parts are connected by either a positive or negative edge (but not both), and hence  $E^{\sim} \cup E^+ \cup E^- = X^2$ .

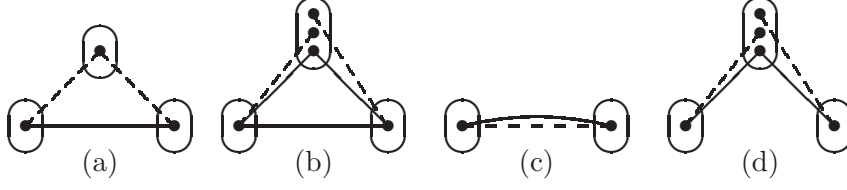


Figure 1: Some example patterns. Points are shown as filled circles, parts as ovals, positive edges as solid lines and negative edges as dashed lines.

**Example 2.4** Some examples of patterns are illustrated in a standard way in Figure 1.

The pattern shown in Figure 1(a) is complete, but the others are not.  $\square$

It will often be convenient to build special patterns to represent binary CSP instances, so we now define the following construction.

**Construction 2.5** For any binary CSP instance  $I = (V, D, C)$ , where  $C = \{R_{uv} \mid u, v \in V, u \neq v\}$ , we define a corresponding complete pattern  $\text{Patt}(I) = (X, E^\sim, E^+, E^-)$  where

- $X = \{x_{v,d} \mid v \in V, d \in D(v)\};$
- $E^\sim = \{(x_{u,a}, x_{v,b}) \mid u = v\};$
- $E^+ = \{(x_{u,a}, x_{v,b}) \mid u \neq v, (a, b) \in R_{uv}\};$
- $E^- = \{(x_{u,a}, x_{v,b}) \mid u \neq v, (a, b) \notin R_{uv}\}.$

We remark that for any instance  $I$  the points of  $\text{Patt}(I)$  are the possible assignments for each individual variable, and the parts of  $\text{Patt}(I)$  correspond to sets of possible assignments for a particular variable. Positive edges in  $\text{Patt}(I)$  correspond to allowed pairs of assignments and are therefore closely related to the edges of the *microstructure* representation of  $I$  defined in [30]; negative edges correspond to disallowed pairs of assignments and are closely related to the edges of the *microstructure complement* discussed in [7].

**Example 2.6** Figure 1(a) shows the pattern  $\text{Patt}(I)$  for a rather trivial instance  $I$  with three variables, each of which has only one possible value. Note that  $I$  has no solution because the only possible assignments for two pairs of variables are in negative edges and hence disallowed by the constraints.  $\square$

A pattern with no positive edges will be called a *negative pattern*. It will sometimes be convenient to build negative patterns from graphs, so we now define the following construction.

**Construction 2.7** For any graph  $G = (V, E)$ , we define a corresponding negative pattern  $\text{Patt}(G) = (X, E^\sim, \emptyset, E^-)$  where

- $X = \{x_{e,v} \mid e \in E, v \in e\};$
- $E^\sim = \{(x_{e,u}, x_{f,v}) \mid u = v\};$
- $E^- = \{(x_{e,u}, x_{f,v}) \mid e = f, u \neq v\}.$

**Example 2.8** Let  $C_3$  be the 3-cycle, that is, the graph with three vertices,  $v_1, v_2, v_3$ , and 3 edges  $e_1, e_2, e_3$ , where  $e_1 = \{v_1, v_2\}, e_2 = \{v_2, v_3\}$  and  $e_3 = \{v_3, v_1\}$ . The associated negative pattern  $\text{Patt}(C_3)$  defined by Construction 2.7 is the pattern with 6 points, 3 parts, and 3 negative edges shown in Figure 2.  $\square$

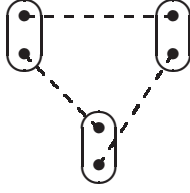


Figure 2: The pattern  $\text{Patt}(C_3)$  constructed from the cycle graph  $C_3$  by Construction 2.7.

In graph theory, a *subdivision* operation on a graph replaces an edge  $(u, v)$  with a path of length two by introducing a new vertex  $z_{uv}$ , and connecting  $u$  to  $z_{uv}$  and  $z_{uv}$  to  $v$  [18]. A graph  $G$  is said to be a topological minor of a graph  $H$  if some sequence of subdivision operations on  $G$  yields a subgraph of  $H$  [18]. We now define an operation on patterns that is analogous to the subdivision operation on graphs, but takes into account the three different types of edges that are present in a pattern. This subdivision operation for patterns is crucial to the idea of defining topological minors in patterns, as described in Section 3.

**Definition 2.9** Let  $P = (X, E^{\sim}, E^+, E^-)$  be a pattern.

For any two distinct parts  $U, V$  of  $P$ , we define  $E_{UV}^+ = E^+ \cap (U \times V)$ ,  $E_{UV}^- = E^- \cap (U \times V)$ , and  $Z_{UV} = \{z_{xy} \mid (x, y) \in E_{UV}^+\} \cup \{z'_{xy}, z''_{xy} \mid (x, y) \in E_{UV}^-\}$ . The subdivision of  $P$  at  $U, V$  is defined to be the pattern  $P_d = (X_d, E_d^{\sim}, E_d^+, E_d^-)$  where

- $X_d = X \cup Z_{UV}$ ;
- $E_d^{\sim} = E^{\sim} \cup (Z_{UV} \times Z_{UV})$ ;
- $E_d^+ = (E^+ \setminus \{(x, y), (y, x) \mid (x, y) \in E_{UV}^+\}) \cup \{(x, z_{xy}), (z_{xy}, x), (z_{xy}, y), (y, z_{xy}) \mid (x, y) \in E_{UV}^+\}$ ;
- $E_d^- = (E^- \setminus \{(x, y), (y, x) \mid (x, y) \in E_{UV}^-\}) \cup \{(x, z'_{xy}), (z'_{xy}, x), (z''_{xy}, y), (y, z''_{xy}) \mid (x, y) \in E_{UV}^-\}$ .

Pattern  $P'$  is called a subdivision of  $P$  if it can be obtained from  $P$  by some (possibly empty) sequence of subdivision operations.

**Example 2.10** The pattern shown in Figure 1(d) can be obtained by performing a single subdivision operation on the pattern shown in Figure 1(c).  $\square$

We remark that positive and negative edges are treated differently in Definition 2.9: a single extra point,  $z_{xy}$ , is added for each positive edge  $(x, y)$ , and *two* extra points,  $z'_{xy}$  and  $z''_{xy}$ , are added for each negative edge (see Example 2.10). This difference reflects a semantic difference between positive and negative edges in a CSP instance, which we illustrate as follows. Suppose that the assignment of  $a$  to variable  $u$  and  $b$  to variable  $v$  extends to a solution. For any other variable  $w$ , the points  $(u, a)$  and  $(v, b)$  must both be compatible with some common point  $(w, c)$ . On the other hand the assignment of  $a$  to variable  $u$  and  $b$  to variable  $v$  may not extend to a solution if there are points  $(w, c)$  and  $(w, d)$  where  $(u, a)$  is incompatible with  $(w, c)$ ,  $(v, b)$  is incompatible with  $(w, d)$  and the rest of the instance forces  $w$  to take either value  $a$  or  $b$ .

### 3 Forbidding patterns

In the remainder of this paper we consider classes of binary CSP instances that are defined by *forbidding* a specified set of patterns from occurring in certain ways, which we now define.

### 3.1 Occurrences of one pattern in another

**Definition 3.1** A pattern  $P_1 = (X_1, E_1^{\sim}, E_1^+, E_1^-)$  is said to have a homomorphism to a pattern  $P_2 = (X_2, E_2^{\sim}, E_2^+, E_2^-)$ , if there is a mapping  $h : X_1 \rightarrow X_2$  such that

- if  $(x, y) \in E_1^{\sim}$  then  $(h(x), h(y)) \in E_2^{\sim}$ , and
- if  $(x, y) \in E_1^+$  then  $(h(x), h(y)) \in E_2^+$ , and
- if  $(x, y) \in E_1^-$  then  $(h(x), h(y)) \in E_2^-$ .

A homomorphism  $h$  from a pattern  $P_1 = (X_1, E_1^{\sim}, E_1^+, E_1^-)$  to a pattern  $P_2 = (X_2, E_2^{\sim}, E_2^+, E_2^-)$  will be said to *preserve parts* if it satisfies the additional property that for all  $(x, y) \in X_1^2$ , if  $(x, y) \notin E_1^{\sim}$ , then  $(h(x), h(y)) \notin E_2^{\sim}$ .

**Definition 3.2** A pattern  $P_1$  is said to occur as a sub-pattern in a pattern  $P_2$ , denoted  $P_1 \xrightarrow{SP} P_2$ , if there is a homomorphism from  $P_1$  to  $P_2$  that preserves parts.

Earlier papers [8, 13] have defined the notions of pattern and the notion of occurring as a sub-pattern in slightly different ways, but these are all essentially equivalent to Definition 3.2.

**Example 3.3** The pattern shown in Figure 1(d) has a homomorphism to the pattern shown in Figure 1(c), but does not occur as a sub-pattern in this pattern. The pattern shown in Figure 1(d) does occur as a sub-pattern in the pattern shown in Figure 1(b).  $\square$

Now we introduce a new form of occurrence that will be our focus in this paper, and will allow us to define a wider range of restricted subproblems of the CSP.

**Definition 3.4** A pattern  $P_1$  is said to occur as a topological minor in a pattern  $P_2$ , denoted  $P_1 \xrightarrow{TM} P_2$ , if some subdivision of  $P_1$  occurs as a sub-pattern in  $P_2$ .

**Example 3.5** The pattern shown in Figure 1(c) occurs as a topological minor in the pattern shown in Figure 1(d) and in the pattern shown in Figure 1(b).  $\square$

**Lemma 3.6** For any patterns  $P, P'$  and  $P''$  the following properties hold:

- (a)  $P \xrightarrow{SP} P$  and  $P \xrightarrow{TM} P$ ;
- (b) If  $P \xrightarrow{SP} P'$ , then  $P \xrightarrow{TM} P'$ ;
- (c) If  $P \xrightarrow{SP} P'$  and  $P' \xrightarrow{SP} P''$ , then  $P \xrightarrow{SP} P''$ ;
- (d) If  $P \xrightarrow{TM} P'$  and  $P' \xrightarrow{TM} P''$ , then  $P \xrightarrow{TM} P''$ .

**Proof:** Part (a) is obtained by taking the identity function as a homomorphism, and an empty sequence of subdivisions. Part (b) is obtained by taking an empty sequence of subdivisions. Part (c) is obtained by composing the two homomorphisms.

Part (d) follows from the following observation: assume that  $h$  is a homomorphism from  $P_1$  to  $P_2$  that preserves parts, and that  $P_3$  is the pattern obtained by performing a subdivision operation on  $P_2$  at parts  $U$  and  $V$ . Now consider the pattern  $Q$  obtained by performing a subdivision operation on  $P_1$  at the parts that are mapped by  $h$  to  $U$  and  $V$ . By our definition of subdivision, it follows that  $h$  can be extended to a homomorphism  $h'$  from  $Q$  to  $P_3$  that preserves parts.

Hence in any sequence of subdivision operations and homomorphisms that preserve parts we can re-order the operations to perform all subdivisions at the start, and then compose all the homomorphisms.  $\blacksquare$



Recall that establishing arc-consistency in an instance  $I$  involves removing domain values from  $I$  and yields the (unique) instance  $\text{AC}(I)$ , hence it cannot introduce an occurrence of a pattern as a sub-pattern or as a topological minor if it did not already occur. This gives the following result.

**Lemma 3.7** *For any patterns  $P$  and  $I$ , where  $I$  represents an instance the following properties hold:*

- (a) *If  $P \xrightarrow{SP} \text{Patt}(\text{AC}(I))$ , then  $P \xrightarrow{SP} \text{Patt}(I)$ ;*
- (b) *If  $P \xrightarrow{TM} \text{Patt}(\text{AC}(I))$ , then  $P \xrightarrow{TM} \text{Patt}(I)$ .*

Establishing arc-consistency can be done in polynomial time, so for many of our results we will only need to consider arc-consistent CSP instances.

### 3.2 Restricted classes of instances

We can use Definition 3.2 to define restricted classes of binary CSP instances by forbidding the occurrence of certain patterns as sub-patterns in those instances.

**Definition 3.8** *Let  $\mathcal{S}$  be a set of patterns.*

*We denote by  $\text{CSP}_{\overline{\text{SP}}}(\mathcal{S})$  the set of all binary CSP instances  $I$  such that for all  $P \in \mathcal{S}$  it is not the case that  $P \xrightarrow{SP} \text{Patt}(I)$ .*

**Definition 3.9** *We will say that a pattern  $P$  is sub-pattern tractable if  $\text{CSP}_{\overline{\text{SP}}}(\{P\})$  is tractable; we will say that a pattern  $P$  is sub-pattern NP-complete if  $\text{CSP}_{\overline{\text{SP}}}(\{P\})$  is NP-complete.*

For simplicity, we write  $\text{CSP}_{\overline{\text{SP}}}(P)$  for  $\text{CSP}_{\overline{\text{SP}}}(\{P\})$ .

The complexity of the class  $\text{CSP}_{\overline{\text{SP}}}(\mathcal{S})$  has been determined for a wide range of patterns [14, 8, 13]. In fact, for all *negative* patterns  $P$  the complexity of  $\text{CSP}_{\overline{\text{SP}}}(P)$  has been completely characterised [8]. To define this characterisation, we need to introduce the idea of *star patterns*.

A connected graph  $G$  is called a *star* if it is acyclic, and has exactly one vertex of degree greater than 2. The vertex of degree greater than 2 in a star graph will be called the central vertex. A pattern  $P$  will be called a *star pattern* if it can be obtained from the pattern  $\text{Patt}(G)$  for some star graph  $G$  by merging zero or more points in the part of  $\text{Patt}(G)$  corresponding to the central vertex of  $G$ .

**Example 3.10** Since the empty graph is a star graph, the simplest star pattern is the empty pattern, which has no points. Some other examples of star patterns are shown in Figure 3.  $\square$

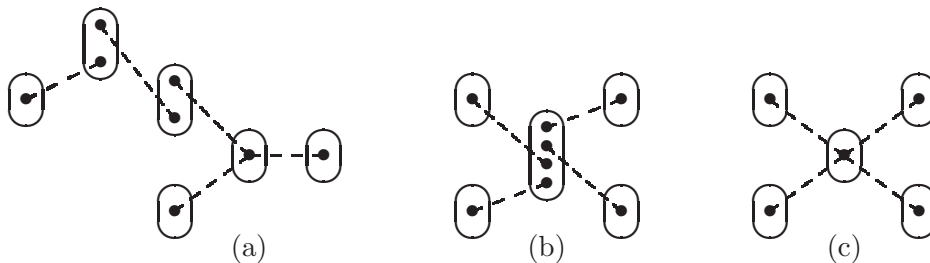


Figure 3: Examples of star patterns.

**Definition 3.11** ([8]) *For any  $k \geq 1$ , the star pattern with 3 branches, each of length  $k$ , where exactly two points are merged in the central part, as shown in Figure 4, is called  $\text{Pivot}(k)$ .*

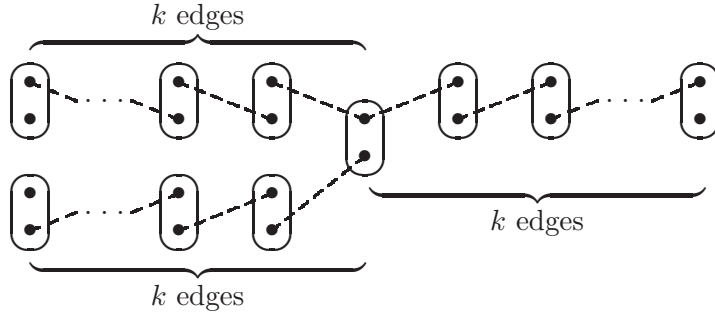


Figure 4: The pattern  $\text{Pivot}(k)$ .

**Theorem 3.12** ([8]) *For any  $k \geq 1$ , the negative pattern  $\text{Pivot}(k)$  shown in Figure 4 is sub-pattern tractable, as are all negative patterns  $P$  such that  $P \xrightarrow{SP} \text{Pivot}(k)$ ; all other negative patterns are sub-pattern NP-complete.*

To go beyond this result and define a wider range of restricted classes we use Definition 3.4 to define restricted classes of binary CSP instances by forbidding the occurrence of certain patterns as topological minors in those instances.

**Definition 3.13** *Let  $\mathcal{S}$  be a set of patterns.*

*We denote by  $\text{CSP}_{\overline{\text{TM}}}(\mathcal{S})$  the set of all binary CSP instances  $I$  such that for all  $P \in \mathcal{S}$  it is not the case that  $P \xrightarrow{TM} \text{Patt}(I)$ .*

**Definition 3.14** *We will say that a pattern  $P$  is topological-minor tractable if  $\text{CSP}_{\overline{\text{TM}}}(\{P\})$  is tractable; we will say that a pattern  $P$  is topological-minor NP-complete if  $\text{CSP}_{\overline{\text{TM}}}(\{P\})$  is NP-complete.*

For simplicity, we write  $\text{CSP}_{\overline{\text{TM}}}(P)$  for  $\text{CSP}_{\overline{\text{TM}}}(\{P\})$ .

By Lemma 3.6 (b), if  $P$  occurs as a sub-pattern of some pattern  $Q$ , then it also occurs as a topological minor of  $Q$ . Hence for any pattern  $P$  we have that  $\text{CSP}_{\overline{\text{TM}}}(P) \subseteq \text{CSP}_{\overline{\text{SP}}}(P)$ . The following is an immediate consequence.

**Lemma 3.15** *If a pattern  $P$  is sub-pattern tractable then  $P$  is also topological-minor tractable.*

**Example 3.16** The two patterns shown in Figure 1(a) and 1(b) are known to be sub-pattern tractable [13, 21]. Hence, they are also topological-minor tractable, by Lemma 3.15.  $\square$

By Lemma 3.6 (d), if  $P$  occurs as a topological minor in  $Q$  then  $\text{CSP}_{\overline{\text{TM}}}(P) \subseteq \text{CSP}_{\overline{\text{TM}}}(Q)$ . The following is an immediate consequence.

**Lemma 3.17** *If pattern  $P \xrightarrow{TM} Q$ , and  $Q$  is topological-minor tractable, then  $P$  is also topological-minor tractable.*

**Example 3.18** We can deduce from Lemma 3.17 that Figure 1(d) is topological-minor tractable, since Figure 1(d) occurs as a sub-pattern (and hence also as a topological minor) in Figure 1(b), and we have already seen that Figure 1(b) is topological-minor tractable.

Similarly, Figure 1(c) is topological-minor tractable, since Figure 1(c) occurs as a topological minor in Figure 1(d). However, Figure 1(c) is sub-pattern NP-complete, since it cannot occur as a sub-pattern of any instance, so for this pattern  $P$ ,  $\text{CSP}_{\overline{\text{SP}}}(P)$  contains all possible CSP instances.  $\square$

The topological-minor tractability of the pattern in Figure 1(c) has a simple corollary.

**Corollary 3.19** *All 2-part patterns are topological-minor tractable.*



**Proof:** Let  $P$  be an arbitrary 2-part pattern with parts  $U$  and  $V$  and let  $Q$  be the pattern in Figure 1(c). The function which maps all of  $U$  to one point of  $Q$  and all of  $V$  to the other point of  $Q$  is necessarily a homomorphism from  $P$  to  $Q$  that preserves parts, since  $Q$  has both a positive and a negative edge between these points, so  $P \xrightarrow{SP} Q$  and hence  $P \xrightarrow{TM} Q$ .

It was shown in Example 3.18 that  $Q$  is topological-minor tractable, so by Lemma 3.17, it follows that  $P$  is topological-minor tractable. ■

For some patterns  $P$ , the sets  $\text{CSP}_{\overline{\text{SP}}}(P)$  and  $\text{CSP}_{\overline{\text{TM}}}(P)$  are identical, as our next result shows. A pattern  $P$  will be called *star-like* if removing the positive edges from  $P$  gives a negative pattern  $P'$  such that  $P' \xrightarrow{SP} P''$  for some star pattern  $P''$ .

**Example 3.20** All of the patterns in Figure 1 and Figure 3 are star-like, but the pattern shown in Figure 2 is not star-like. □

**Proposition 3.21** *If  $P$  is a star-like negative pattern, then  $\text{CSP}_{\overline{\text{TM}}}(P) = \text{CSP}_{\overline{\text{SP}}}(P)$ .*

**Proof:** By Lemma 3.6 (b), for any pattern  $P$  we have that  $\text{CSP}_{\overline{\text{TM}}}(P) \subseteq \text{CSP}_{\overline{\text{SP}}}(P)$ .

To obtain the reverse inclusion, let  $P$  be a star-like negative pattern, and let  $Q$  be a star pattern such that  $P \xrightarrow{SP} Q$ . By the definition of star pattern, for any subdivision  $Q'$  of  $Q$ , we have that  $Q \xrightarrow{SP} Q'$ . Hence, by Lemma 3.6 (c)  $P \xrightarrow{SP} Q'$ , so  $\text{CSP}_{\overline{\text{SP}}}(P) \subseteq \text{CSP}_{\overline{\text{SP}}}(Q')$ . But this implies, by Definition 3.4, that  $\text{CSP}_{\overline{\text{SP}}}(P) \subseteq \text{CSP}_{\overline{\text{TM}}}(P)$ . ■

**Example 3.22** By Theorem 3.12, any pattern  $\text{Pivot}(k)$  is sub-pattern tractable, and by Proposition 3.21 we know that forbidding  $\text{Pivot}(k)$  as a topological minor defines the same set of instances as forbidding  $\text{Pivot}(k)$  as a sub-pattern. Therefore, for any  $k \geq 1$ , the pattern  $\text{Pivot}(k)$  is also topological-minor tractable.

Similarly, by Theorem 3.12, each star pattern  $P$  shown in Figure 3 is sub-pattern NP-complete. By Proposition 3.21, for each of these patterns  $\text{CSP}_{\overline{\text{TM}}}(P) = \text{CSP}_{\overline{\text{SP}}}(P)$ . Consequently, these patterns are also topological-minor NP-complete. □

We now give a partial converse of Proposition 3.21, by showing that for all patterns  $P$  that are not star-like,  $\text{CSP}_{\overline{\text{TM}}}(P)$  cannot be expressed by forbidding any finite set of sub-patterns. This means that the notion of forbidding the occurrence of a pattern as a topological minor provides more expressive power than forbidding arbitrary (finite) sets of patterns from occurring as sub-patterns.

**Proposition 3.23** *If  $P$  is a pattern that is not star-like, then  $\text{CSP}_{\overline{\text{TM}}}(P) \neq \text{CSP}_{\overline{\text{SP}}}(\mathcal{S})$  for all finite sets of patterns  $\mathcal{S}$ .*

**Proof:** Let  $P$  be a pattern that is not star-like, and let  $P'$  be the negative pattern obtained by removing all positive edges of  $P$ . Note that  $P' \xrightarrow{SP} P$ .

In any pattern, say that a part  $U$  is *distinguished* if two negative edges share a single point in  $U$  or if there are negative edges from  $U$  to more than two other parts.

Since  $P$  is not star-like, the negative pattern  $P'$  must contain a cycle of parts connected by negative edges, or two distinguished parts.

Hence, for any fixed  $k$ , by a sufficiently long sequence of subdivision operations, we can construct a subdivision  $P''$  of  $P'$  which either has a cycle of parts of length greater than  $k$  or two distinguished parts separated by a sequence of connected parts of length greater than  $k$ . By adding positive edges, we can then convert  $P''$  into a complete pattern of the form  $\text{Patt}(I)$  for some CSP instance  $I$ .

Now for any fixed finite set of patterns  $\mathcal{S}$  there will be a bound  $k$  on the number of parts of any pattern in  $\mathcal{S}$ . It follows that  $\text{CSP}_{\overline{\text{TM}}}(P)$  cannot be defined by forbidding the sub-patterns in  $\mathcal{S}$ , since  $I \notin \text{CSP}_{\overline{\text{TM}}}(P)$  but no pattern in  $\mathcal{S}$  can occur as a sub-pattern in  $\text{Patt}(I)$ . ■

## 4 Structural restrictions

For any CSP instance  $I = (V, D, C)$ , the *constraint graph* of  $I$  is defined to be the graph  $(V, E)$ , where  $E$  is the set of pairs  $\{x, y\}$  for which the associated constraint  $R_{xy}$  is non-trivial. A number of tractable subproblems of the CSP have been defined by specifying restrictions on the constraint graph; such restricted classes of instances are known as *structural classes* [26, 34].

It is known that a structural class of binary CSP instances is tractable if and only if every instance has a constraint graph of bounded treewidth [26, Theorem 5.1] (subject to the standard complexity-theoretic assumption that  $\text{FPT} \neq \text{W}[1]$ , which we will assume throughout this section [20, 23]). We show in this section that such classes cannot be defined by forbidding the occurrence of a finite set of sub-patterns. However, they *can* be defined by forbidding the occurrence of one or more patterns as topological minors.

We will also use this characterisation of tractable structural classes to show that a large class of negative patterns are topological minor tractable.

First we extend the notion of a constraint graph to arbitrary patterns.

**Definition 4.1** *For any pattern  $P$ , the constraint graph of  $P$ , denoted  $\text{CG}(P)$ , is defined to be the graph  $(V, E)$ , where  $V$  is the set of all parts of  $P$ , and  $E$  is the set of pairs of parts  $\{U, W\}$  such that there is a negative edge  $(x, y) \in P$  with  $x \in U$  and  $y \in W$ .*

For any binary CSP instance  $I$ , the constraint graph of  $I$  is given by  $\text{CG}(\text{Patt}(I))$ .

Now we note the close link between our notion of a pattern occurring as a topological minor of another pattern and the standard notion of a topological minor in a graph [18].

**Lemma 4.2** *For any graph  $G$  and any pattern  $P$ ,  $\text{Patt}(G) \xrightarrow{TM} P$  if and only if  $G$  is a topological minor of the graph  $\text{CG}(P)$ .*

The simplest structural class of CSP instances of bounded treewidth is the class of instances whose constraint graph is acyclic (that is, has treewidth 1). This class was one of the first sub-problems of the CSP to be shown to be tractable [24]. We now show that this class can be characterised very simply by excluding the single pattern  $\text{Patt}(C_3)$  shown in Figure 2 from occurring as a topological minor.

**Proposition 4.3** *The class of acyclic binary CSP instances equals  $\text{CSP}_{\overline{\text{TM}}}(\text{Patt}(C_3))$ .*

**Proof:** The class of acyclic graphs may be characterised as graphs which do not contain  $C_3$  as a topological minor [18]. Hence, by Lemma 4.2 and Definition 4.1, a binary CSP instance  $I$  is acyclic if and only if it is not the case that  $\text{Patt}(C_3) \xrightarrow{TM} \text{Patt}(I)$ . ■

Since the pattern  $\text{Patt}(C_3)$  is not star-like (see Example 3.20), it follows immediately from Proposition 3.23 that acyclic CSP instances cannot be defined by any finite set of forbidden sub-patterns.

**Corollary 4.4** *The class of acyclic binary CSP instances is not equal to  $\text{CSP}_{\overline{\text{SF}}}(\mathcal{S})$  for any finite set of patterns  $\mathcal{S}$ .*

Proposition 4.3 can easily be extended to any of the tractable classes of binary CSP instances defined by imposing any fixed bound on the treewidth of the constraint graph [25], although in this case the set of forbidden patterns is explicitly known only for  $k \leq 3$  [1].

**Theorem 4.5** *For any fixed  $k \geq 1$ , the class of binary CSP instances with constraint graphs of treewidth at most  $k$  equals  $\text{CSP}_{\overline{\text{TM}}}(\mathcal{S}_k)$  for some finite set of patterns  $\mathcal{S}_k$ .*

**Proof:** The graph minor theorem [36] implies that for any fixed  $k \geq 1$  there is a finite set  $O_k$  of graphs such that the class of graphs of treewidth at most  $k$  is precisely the class of graphs excluding all graphs from the set  $O_k$  as topological minors [18]. (More precisely, the graph minor theorem gives a finite set of minors as obstructions but this set can be turned into a finite set of topological minors as obstructions in a standard way, see [18, Exercise 34, Chapter 12].) Consequently, by Lemma 4.2, for any  $k \geq 1$  the class of binary CSP instances with constraint graphs of treewidth at most  $k$  can be defined as  $\text{CSP}_{\overline{\text{TM}}}(\mathcal{S}_k)$  for the finite set of negative patterns  $\mathcal{S}_k$  given by  $\mathcal{S}_k = \{\text{Patt}(G) \mid G \in O_k\}$ . ■

In fact, we are able to show that many other patterns are topological-minor tractable using other standard results from graph theory. The following theorem characterises the topological-minor tractability of patterns of the form  $\text{Patt}(G)$ , for all graphs  $G$  of maximum degree three.

**Theorem 4.6** *Let  $G$  be an arbitrary graph of maximum degree three. Then,  $\text{Patt}(G)$  is topological-minor tractable if and only if  $G$  is planar (assuming  $\text{FPT} \neq \text{W}[1]$ ).*

**Proof:** One of the well-known results of Robertson and Seymour shows that the class of graphs obtained by excluding  $G$  as a minor has bounded treewidth if and only if  $G$  is planar [35] (see also [18, Theorem 12.4.3]). It is known that for a graph  $G$  of maximum degree three and any graph  $G'$ ,  $G$  is a minor of  $G'$  if and only if  $G$  is a topological minor of  $G'$  [18, Proposition 1.7.4 (ii)]. Thus, for a graph  $G$  of maximum degree three, the class of graphs obtained by excluding  $G$  as a topological minor has bounded treewidth if and only if  $G$  is planar. The theorem then follows from the fact that, assuming  $\text{FPT} \neq \text{W}[1]$ , a structural class of binary CSP instances is tractable if and only if the associated class of constraint graphs is of bounded treewidth [26]. ■

Unfortunately this result does not extend to graphs of higher degree, as the following example shows.

**Example 4.7** Consider a star graph  $G$  where the central vertex has degree 4. Note that  $G$  is planar.

In all subdivisions of  $G$ , the central vertex still has degree 4, so it cannot occur as a topological minor in any graph of maximum degree three. Hence, by Lemma 4.2,  $\text{Patt}(G)$  cannot occur as a topological minor in any CSP instance whose constraint graph is a hexagonal grid. Since the treewidth of the class of hexagonal grids is unbounded [18], this structural class of CSP instances is intractable, assuming  $\text{FPT} \neq \text{W}[1]$ , by the results of [26]. □

## 5 Tractable classes that generalise acyclicity

In this section we will give several more examples of patterns that are topological-minor tractable. We conclude the section with Theorem 5.3 where we define several new tractable classes which properly extend the class of acyclic CSP instances discussed in Section 4.

Consider the patterns shown in Figure 5. By Theorem 3.12,  $J$  is sub-pattern tractable and hence also topological-minor tractable, by Lemma 3.15. However, the remaining patterns,  $K$  and  $L$  are more interesting.

**Theorem 5.1** *The pattern  $K$ , shown in Figure 5, is sub-pattern NP-complete but topological-minor tractable.*

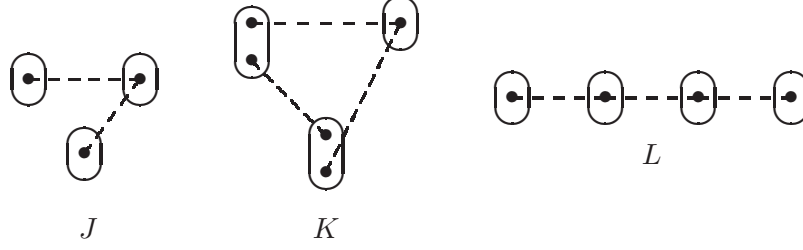


Figure 5: Three patterns which are topological-minor tractable.

**Proof:** By Theorem 3.12,  $K$  is sub-pattern NP-complete.

Now consider an instance  $I$  in which the pattern  $K$  does not occur as a topological minor, and let  $G_I$  be the constraint graph of  $I$ . Suppose the pattern  $J$ , shown in Figure 5, occurs as a sub-pattern on the triple of variables  $(x, y, z)$  in  $I$ , with  $y$  being the variable at which the two negative edges meet.

Since  $K$  does not occur as a topological minor in  $I$ , it follows that there is no path from  $x$  to  $z$  in  $G_I$  that does not pass through  $y$ . We can therefore find a tree-decomposition of  $G_I$  into components in which  $J$  does not occur as a sub-pattern, where these components overlap only at such variables  $y$ , which we can call articulation variables.

Since  $\text{CSP}_{\overline{\text{SP}}}(J)$  is tractable, as noted above, any subinstance corresponding to a leaf component can be solved in polynomial time for each possible assignment to the unique articulation variable which joins it to its parent component in the tree-decomposition. This leads to the elimination of the leaf component and possible elimination of some values in the domain of this articulation variable. The original instance  $I$  can be solved in polynomial time by repeatedly solving and eliminating sub-instances corresponding to leaf components in this way. ■

We will show in Theorem 5.2 below that the pattern  $L$  shown in Figure 5 is also topological-minor tractable. In order to do so, we will extend the proof technique used in Theorem 5.1 to a *generic scheme* for proving topological-minor tractability of patterns.

To develop our generic scheme we need some results from graph theory. If  $T$  is a set of vertices of a graph  $G$ , we write  $G[T]$  for the induced graph on  $T$ . We say that  $(T_1, T_2)$  is a *separation* of  $G$  if  $G = G[T_1] \cup G[T_2]$ . The *separator* of the separation  $(T_1, T_2)$  is  $T_1 \cap T_2$  and its *order* is  $|T_1 \cap T_2|$ . The *torso* of  $T_1$  in the separation  $(T_1, T_2)$  is obtained from the induced graph  $G[T_1]$  by adding every edge between the vertices of the separator. A *Tutte decomposition* of a graph  $G$  is a tree, where each node is labelled with a subset of the vertices of  $G$ , each arc induces a separation of  $G$  of order at most two, and the torso of each node is three-connected, or a cycle, or has at most 2 vertices. It is known that every finite graph has a Tutte decomposition of this kind [39].

To demonstrate topological-minor tractability for a pattern  $P$  we proceed as follows. Let  $I$  be an instance in which  $P$  does not occur as a topological minor and let  $G_I$  be its constraint graph. We denote by  $n$  the number of variables in  $I$  and by  $d$  the maximum domain size of any variable in  $I$ .

Build a Tutte decomposition of  $G_I$ , and consider any leaf node  $S$  in this decomposition, inducing the separation  $(S, T)$  of  $G_I$ . Let  $I[S]$  be the subinstance of  $I$  on the variables of  $S$ . Suppose that the following two assumptions hold:

- (A1)  $I[S]$  can be solved and its solutions projected onto the separator of  $(S, T)$  in polynomial time; the resulting reduced instance on  $T$  will be denoted by  $I'[T]$ .
- (A2)  $P$  does not occur as a topological minor in  $\text{Patt}(I'[T])$ .

Then it follows that a recursive algorithm, which at each step chooses some leaf  $S$  of the decomposition, and then solves the associated sub-problem  $I[S]$  to obtain the reduced instance

$I'[T]$ , will solve the original instance using a polynomial (in  $n$  and  $d$ ) number of calls to the polynomial-time algorithm from (A1).

In the proofs below we will omit the simple cases where the separator has order less than two, or  $S$  contains at most 3 vertices, or the torso of  $S$  is a cycle (and hence has treewidth 2 and is solvable in polynomial time). Hence we will assume that the torso of  $S$  contains more than three vertices and is three-connected.

Finally, note that if  $\{u, v\}$  is an order-2 separator for some separation  $(S, T)$  of  $G_I$  then we have the following:

- Any path in  $G_I$  from a vertex in  $S$  to a vertex in  $T$  must pass through  $u$  or  $v$ ;
- There must exist some path from  $u$  to  $v$  in  $G_I[T]$ , which we will denote  $path_T(u, v)$ .

We now use this generic scheme to prove the tractability of pattern  $L$  from Figure 5.

**Theorem 5.2** *The pattern  $L$ , shown in Figure 5, is sub-pattern NP-complete but topological-minor tractable.*

**Proof:** By Theorem 3.12,  $L$  is sub-pattern NP-complete.

To establish topological-minor tractability using the generic scheme we only need to establish the two assumptions.

(A1) Let  $J$  be the pattern consisting of two intersecting negative edges, shown in Figure 5. Suppose that  $J$  occurs in  $\text{Patt}(I[S])$  as a sub-pattern on two *disjoint* triples of variables  $(x, y, z)$  and  $(x', y', z')$  in  $I[S]$ . As explained above for the generic scheme, we can assume that the torso of  $S$  is 3-connected. It follows by Menger's theorem [19] that there are three disjoint paths from  $x$  to  $x'$  in the torso of  $S$ . There must be one of these paths,  $\pi$ , which does not pass through  $y$  or  $y'$ . Hence there must be a subpath  $\sigma$  of  $\pi$  which begins at  $x$  or  $z$  and ends at  $x'$  or  $z'$  (or vice versa, i.e., which begins at  $x'$  or  $z'$  and ends at  $x$  or  $z$ ) and which does not pass through any other variables in  $\{x, y, z, x', y', z'\}$ . Without loss of generality, suppose that  $\sigma$  joins  $x$  to  $x'$ . But then  $L$  occurs as a topological minor on the extended path  $\sigma^+$  given by  $z \rightarrow y \rightarrow x, \sigma, x' \rightarrow y' \rightarrow z'$ .

But this implies that  $L$  occurs as a topological minor in  $\text{Patt}(I)$ , since if  $\sigma^+$  passes by the edge  $\{u, v\}$  in the torso of  $S$ , this edge can be replaced by  $path_T(u, v)$  which is a path from  $u$  to  $v$  in  $T$ , whose existence was noted in the discussion above. Since this contradicts our initial assumption, we can deduce that  $J$  does not occur in  $\text{Patt}(I[S])$  as a sub-pattern on two *disjoint* triples.

We can therefore deduce that all pairs of triples of variables  $(x, y, z), (x', y', z')$  for which  $J$  occurs as a sub-pattern in  $\text{Patt}(I[S])$  intersect, i.e.  $\{x, y, z\} \cap \{x', y', z'\} \neq \emptyset$ . Now, consider an arbitrary triple of variables  $(x, y, z)$  on which  $J$  occurs as a sub-pattern. It follows that the instance which results after any instantiation (and removal) of the three variables  $x, y, z$  contains no occurrence of  $J$  as a sub-pattern, since for each triple of variables  $(x', y', z')$  on which  $J$  occurs in  $I[S]$ , at least one of its variables has been eliminated by instantiation.

Thus, after instantiation of at most three variables,  $\text{Patt}(I[S])$  does not contain  $J$  as a sub-pattern. This also holds for any version of  $I[S]$  obtained by instantiating the variables  $u, v$ . As noted above,  $\text{CSP}_{\overline{\text{SP}}}(J)$  is tractable. We can therefore determine in polynomial time which instantiations of  $u, v$  can be extended to a solution of  $I[S]$ . We remove the pair  $(p, q)$  from  $R_{uv}$  in  $I$  whenever the assignment of  $p$  to  $u$  and  $q$  to  $v$  cannot be extended to a solution to  $I[S]$ . Finally, we delete all variables in  $S$  from  $I$  apart from  $u$  and  $v$ . Proceeding in this way we construct  $I'[T]$  in polynomial time as required.

(A2): Suppose, for a contradiction, that we introduce some occurrence of the pattern  $L$  as a topological minor in  $\text{Patt}(I'[T])$  when reducing  $I$  to  $I'[T]$ . This occurrence of  $L$  must use a newly-introduced edge in  $I'[T]$ . During the reduction from  $I$  to  $I'[T]$ , we can introduce negative (but not positive) edges in  $\text{Patt}(I'[T])$  between the parts corresponding to  $u$  and  $v$ . Suppose

that a negative edge  $(p, q)$  is introduced by the reduction from  $I$  to  $I'[T]$ . This can only be the case if there was a path  $\pi = (u, w_1, \dots, w_t, v)$  in the constraint graph  $G_I[S]$  and hence a sequence of negative edges between the corresponding parts in  $\text{Patt}(I[S])$  linking  $p$  to  $q$ . This means that we can replace the newly-introduced edge in the occurrence of  $L$  in  $\text{Patt}(I'[T])$  by a sequence of negative edges so that  $L$  occurs as a topological minor in  $\text{Patt}(I)$  for the original instance  $I$ . This contradiction shows that we cannot introduce  $L$  as a topological minor in  $\text{Patt}(I'[T])$  when reducing  $I$  to  $I'[T]$ .

Hence we have established both assumptions, so the result follows by our generic proof scheme. Note that the number of instances of  $\text{CSP}_{\overline{\text{SP}}}(J)$  that need to be solved is  $O(nd^5)$ . ■

As our final result in this section we show how the well-known tractable class of acyclic instances can be generalised to obtain larger tractable classes defined by forbidding the occurrence of certain patterns as topological minors. The main tool we use will again be the generic scheme based on Tutte decompositions described above.

**Theorem 5.3** *Let  $P_0$  be any sub-pattern tractable pattern with three parts,  $U_1, U_2, U_3$  where there is at most one negative edge between  $U_1$  and  $U_2$ , and between  $U_2$  and  $U_3$ , and no edges between  $U_1$  and  $U_3$ .*

*Let  $P$  be a pattern with four parts  $U_1, U_2, U_3, U_4$  obtained by extending  $P_0$  as follows. The pattern  $P$  has six new points  $p_1, p_2 \in U_1$ ,  $q_1, q_2 \in U_4$ , and  $r_1, r_2 \in U_3$ , together with three new negative edges  $\{p_1, r_1\}$ ,  $\{p_2, q_1\}$ ,  $\{q_2, r_2\}$  (see Figure 6). Any such  $P$  is topological-minor tractable.*

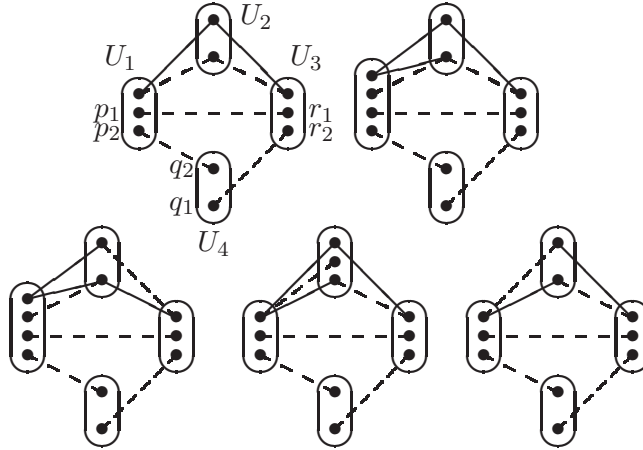


Figure 6: Topological-minor tractable patterns derived from sub-pattern tractable patterns.

**Proof:** The proof uses the generic scheme described in Section 5, so we only need to establish the two assumptions.

(A1) Suppose first that  $P_0$  occurs as a sub-pattern in  $\text{Patt}(I[S])$  on the triple of variables  $(x, y, z)$ . As explained above, when using the generic scheme we will assume that the torso of  $S$  is three-connected. Then, by Menger's theorem there are three disjoint paths  $\pi_1, \pi_2, \pi_3$  from  $x$  to  $z$  in the torso of  $S$ . Hence there must be two of these paths, say  $\pi_1$  and  $\pi_2$ , which do not pass through  $y$ . But this implies that  $P$  occurs as a topological minor in  $\text{Patt}(I)$ , since if either  $\pi_1$  or  $\pi_2$  passes through the edge  $\{u, v\}$  in the torso of  $S$ , this edge can be replaced by  $\text{path}_T(u, v)$  which is a path from  $u$  to  $v$  in  $G_I[T]$ , whose existence was shown in the discussion of the generic scheme above. Since this contradicts our initial assumption, we can assume that  $P_0$  does *not* occur as a sub-pattern in  $\text{Patt}(I[S])$ . This also holds for any sub-problem of  $I[S]$  obtained by instantiating the variables  $u, v$ . Therefore, by the sub-pattern tractability of  $P_0$ , we can determine in polynomial time which instantiations of  $u, v$  can be extended to a solution



of  $I[S]$ . We remove the pair  $(p, q)$  from  $R_{uv}$  in  $I$  whenever the assignment of  $p$  to  $u$  and  $q$  to  $v$  cannot be extended to a solution to  $I[S]$ . Finally, we delete all variables in  $S$  from  $I$  except for  $u$  and  $v$ . Proceeding in this way we construct  $I'[T]$  in polynomial time, as required.

**(A2)** Suppose, for a contradiction, that we introduce the pattern  $P$  as a topological minor of  $\text{Patt}(I'[T])$  when reducing  $I$  to  $I'[T]$ . This occurrence of  $P$  must use a newly-introduced negative edge. Observe that, by definition,  $P$  contains at most one negative edge between any two parts. Suppose that a negative edge  $(p, q)$  is introduced by the reduction from  $I$  to  $I'[T]$ . This can only be the case if there was a path  $\pi = (u, w_1, \dots, w_t, v)$  in the constraint graph  $G_I[S]$  and hence a sequence of negative edges between the corresponding parts in  $\text{Patt}(I[S])$  linking  $p$  to  $q$ . Furthermore, in  $I'[T]$ , if there is a positive edge  $(p', q')$  between the parts corresponding to  $u$  and  $v$  then there is necessarily a solution to  $I[S]$  including the assignments  $p'$  to  $u$  and  $q'$  to  $v$  (and hence a solution on the subinstance  $I[\pi]$  of  $I[S]$  on the path  $\pi = (u, w_1, \dots, w_t, v)$  in  $I[S]$ ). This means that we can replace the edge  $(p, q)$  in the occurrence of  $P$  in  $I'[T]$  by a sequence of negative edges so that  $P$  occurs as a topological minor in  $\text{Patt}(I)$  for the original instance  $I$ . This contradiction shows that we cannot introduce an occurrence of  $P$  as a topological minor in  $\text{Patt}(I'[T])$  by reducing  $I$  to  $I'[T]$ .

Hence we have established both assumptions, so the result follows by our generic proof scheme. Note that the number of instances of  $\text{CSP}_{\overline{\text{SP}}}(P_0)$  that need to be solved is  $O(nd^2)$ . ■

Theorem 5.3 tells us that any pattern  $P_0$  satisfying the conditions of the theorem can be used as a building block for a topological-minor tractable pattern by adding two paths of negative edges. The resulting topological-minor tractable patterns for all such patterns  $P_0$  are shown in Figure 6. For each of these patterns  $P$ , the pattern shown in Figure 2 occurs as a sub-pattern and hence as a topological minor of  $P$ . Thus, by the transitivity of occurrence as a topological minor, each tractable class  $\text{CSP}_{\overline{\text{TM}}}(P)$  necessarily contains all acyclic binary CSP instances.

All of the tractable classes identified in this section rely on finding a Tutte decomposition of the constraint graph of an instance and showing that a certain pattern does not occur as a sub-pattern in any node of the decomposition. Since detecting whether a fixed pattern occurs as a sub-pattern can be done in polynomial time [8], and constructing a Tutte decomposition is also polynomial-time, all the tractable classes described in this section can be identified in polynomial-time.

## 6 Detection of topological minors

For every fixed undirected graph  $H$ , there is an  $O(n^3)$  time algorithm that tests, given a graph  $G$  with  $n$  vertices, if  $H$  is a topological minor of  $G$  [27].

However, for detecting topological minors in patterns the situation is different. For some patterns  $P$  it is possible to decide in polynomial time whether  $P$  occurs as a topological minor of a given pattern  $P'$ . For example, by Lemma 4.2, deciding whether a negative pattern of the form  $\text{Patt}(G)$  for some graph  $G$  occurs as a topological minor in a pattern  $P'$  amounts to detecting whether  $G$  is a topological minor of the constraint graph of  $P'$ , and hence can be achieved in polynomial time [27]. By Proposition 3.21, deciding whether a star-like negative pattern occurs as a topological minor in an instance can also be achieved in polynomial time because this is equivalent to deciding whether it occurs as a sub-pattern, which is achievable in polynomial time by exhaustive search. It follows from our observations at the end of Section 5 that deciding whether any of the patterns shown in Figure 5 or Figure 6 occur as a topological minor in an instance can also be done in polynomial time.

However, for some patterns (such as the 4-part pattern  $M$  shown in Figure 7), it is coNP-complete to determine whether the pattern occurs as a topological minor in an arbitrary given pattern, as our next results show.

Characterising all patterns  $P$  for which it is possible to decide in polynomial time whether  $P$  occurs as a topological minor in a given pattern  $P'$  remains an open problem.

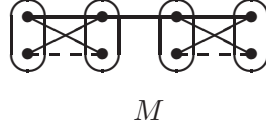


Figure 7: A pattern that is coNP-complete to detect as a topological minor.

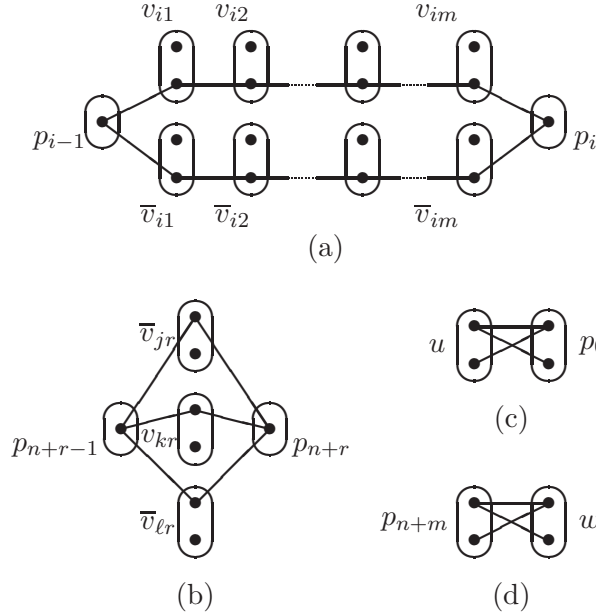


Figure 8: The building blocks for the CSP instance  $I$  constructed in the proof of Theorem 6.1.

**Theorem 6.1** *The problem of deciding  $I \in \text{CSP}_{\overline{\text{TM}}}(M)$  is coNP-complete.*

**Proof:** The problem is clearly in coNP, so it suffices to give a reduction from 3-SAT to the complement of the problem of deciding  $I \in \text{CSP}_{\overline{\text{TM}}}(M)$ .

Let  $I_{SAT}$  be an instance of 3-SAT with variables  $x_1, \dots, x_n$  and clauses  $C_1, \dots, C_m$ . We will create a binary CSP instance  $I$  with variables  $\{u, w\} \cup \{p_i \mid i = 0 \dots n + m\} \cup \{v_{ir}, \bar{v}_{ir} \mid i = 1 \dots n, r = 1 \dots m\}$ , such that determining whether  $M \xrightarrow{TM} \text{Patt}(I)$  is equivalent to deciding whether  $I_{SAT}$  has a solution.

Consider the patterns shown in Figure 8 where each part is labelled with a variable of  $I$ . Connect with negative edges all pairs of points in different parts in these patterns that are not already connected, to obtain complete patterns representing sub-instances of  $I$ . The instance  $I$  is constructed by combining sub-instances of these four kinds, as follows:

- For each variable  $x_i$  in  $I_{SAT}$  we include an instance  $I_{x_i}$  obtained from a pattern of the form shown in Figure 8(a).
- For each clause  $C_r$  in  $I_{SAT}$  we include an instance  $I_{C_r}$  obtained from a pattern of the form shown in Figure 8(b), where the choice of variables for the three central parts depends on the literals in the clause  $C_r$  in the following way: variable  $v_{ir}$  corresponds to  $\neg x_i$

occurring in clause  $C_r$  and variable  $\bar{v}_{ir}$  corresponds to  $x_i$  occurring in clause  $C_r$ . That is, the example shown in Figure 8(b) would correspond to the clause  $x_j \vee \neg x_k \vee x_\ell$ .

- We also include one instance obtained from the pattern shown in Figure 8(c) and one instance obtained from the pattern shown in Figure 8(d);
- Finally, we complete the resulting pattern by adding negative edges to obtain  $\text{Patt}(I)$ .

The only pairs of parts in  $\text{Patt}(I)$  that are connected by more than one positive edge are  $\{u, p_0\}$  and  $\{p_{n+m}, w\}$ . So, if  $M$  occurs as a topological minor in  $\text{Patt}(I)$ , then the points of  $M$  must map injectively to these two pairs of parts. Therefore, deciding whether  $M$  occurs as a topological minor in  $\text{Patt}(I)$  is equivalent to deciding whether there is a path  $\pi$  of positive edges from  $p_0$  to  $p_{n+m}$  in  $\text{Patt}(I)$  which passes through each part at most once.

Any such path  $\pi$  must pass through the points  $p_0, p_1, \dots, p_{n+m}$  in this order, because the positive edges in  $\text{Patt}(I_{x_i})$  ( $1 \leq i \leq n$ ) use different points in each part (shown as the bottom of the two points in Figure 8) from the positive edges in  $\text{Patt}(I_{C_r})$  ( $1 \leq j \leq m$ ) (which use the top points), so there are no short-cuts.

If such a path  $\pi$  exists, then for each variable  $x_i$  of  $I_{SAT}$ , the path  $\pi$  must select in  $I_{x_i}$  either the upper path through variables  $v_{ir}$  ( $r = 1, \dots, m$ ) or the lower path through variables  $\bar{v}_{ir}$  ( $r = 1, \dots, m$ ). Thus  $\pi$  selects a truth value for each variable  $x_i$ : TRUE if  $\pi$  follows the upper path of these two paths, FALSE otherwise.

Moreover, for each clause  $C_r$  in  $I_{SAT}$  the path  $\pi$  must pass from  $p_{n+r-1}$  to  $p_{n+r}$  by one of the three paths in  $\text{Patt}(I_{C_r})$  without passing through parts that have been already used by  $\pi$ . Thus, for  $\pi$  to exist it must have already assigned TRUE to one of the literals of the clause  $C_r$ .

It follows that  $M$  occurs as a topological minor of  $\text{Patt}(I)$  if and only if  $\text{Patt}(I)$  has an appropriate path of positive edges, which occurs if and only if  $I_{SAT}$  is satisfiable. ■

The instance  $I$  in the proof of Theorem 6.1 is clearly inconsistent since there are some constraint relations which are empty. An instance is said to be *globally consistent* if each variable-value assignment  $(v_i, a)$  can be extended to a solution. We now give another example of a pattern which is coNP-complete to detect as a topological minor even in globally-consistent instances.

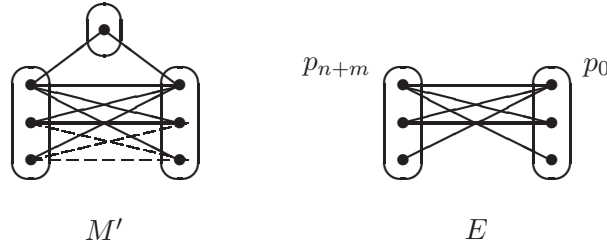


Figure 9: The pattern  $M'$  and one of the building blocks for the globally-consistent instance  $I'$  in which detecting it is coNP-complete.

**Theorem 6.2** *The problem of deciding  $I \in \text{CSP}_{\overline{\text{TM}}}^{\text{TM}}(M')$  for globally-consistent instances  $I$  is coNP-complete.*

**Proof:** We use a very similar construction to the one used in the proof of Theorem 6.1. Let  $I$  be the instance constructed in that proof. Let  $I'$  be identical to  $I$  except that:

- we replace the sub-instances obtained from the patterns shown in Figure 8(c) and Figure 8(d) with a single sub-instance obtained from the pattern  $E$  shown in Figure 9;

- for each variable-value assignment  $(v, a)$  of  $I$ , we create a solution which is an extension of  $(v, a)$ , by adding a new value  $b(v, a, v')$  to the domain of each variable  $v' \neq v$  which is compatible with  $(v, a)$  and with all such values  $b(v, a, v'')$  ( $v'' \notin \{v, v'\}$ ), but incompatible with all other variable-value assignments.

By construction,  $I'$  is clearly globally-consistent. If  $M'$  occurs as a topological minor of  $\text{Patt}(I')$ , then the points of  $M'$  must map injectively to the points of  $E$ , and so again the question is whether there is a path (of length greater than 1) of positive edges linking  $p_0$  to  $p_{n+m}$ . As in the proof of Theorem 6.1, this path exists if and only if the instance  $I_{SAT}$  is satisfiable. Hence, the decision problem  $I \in \text{CSP}_{\overline{\text{TM}}}(P_X)$  for globally-consistent instances  $I$  is coNP-complete. ■

Theorems 6.1 and 6.2 show that not all classes defined by forbidding topological minors can be recognized in polynomial time. Certain uses of tractable classes require polynomial-time recognition: in particular, the automatic recognition and resolution of easy instances within general-purpose solvers. On the other hand, polynomial-time recognition of a tractable class  $\mathcal{C}$  is not required for the construction of a polynomial-time solvable relaxation in  $\mathcal{C}$ , nor in the proof (by a human being) that a subproblem of CSP encountered in practice falls in  $\mathcal{C}$ .

## 7 Augmented patterns

For some CSP instances we have extra information such as an ordering on the variables or on the domains (or both). In this section we introduce the idea of adding an additional relation to a pattern to allow us to capture information of this kind. A pattern  $P$ , together with an additional relation on the points of  $P$  will be called an *augmented pattern*. We will demonstrate that augmented patterns can be used to define new hybrid tractable classes that extend those described in earlier sections.

**Definition 7.1** *An augmented pattern is a pair  $(P, R)$  where  $P$  is a pattern and  $R$  is a relation (of any arity) over the points of  $P$ . The augmented pattern  $(P, R)$  will be denoted  $P_R$ .*

Obvious examples of relations that could be added to a pattern are disequality relations or partial orders on points, and this idea has been explored in a number of papers [8, 14, 15].

**Definition 7.2** *A homomorphism between augmented patterns  $P_R$  and  $P'_{R'}$  is a homomorphism  $h$  from  $P$  to  $P'$  such that for all tuples  $(x_1, x_2, \dots, x_k) \in R$ , the tuple  $(h(x_1), h(x_2), \dots, h(x_k)) \in R'$ .*

Using this extended definition of homomorphism, we can extend the notion of occurring as a sub-pattern (Definition 3.2) and occurring as a topological minor (Definition 3.4) to augmented patterns in the natural way.

Now we can extend Definitions 3.8 and 3.13, as follows, to define restricted classes of CSP instances and associated relations by forbidding the occurrence of certain augmented patterns.

**Definition 7.3** *Let  $m$  be a constant, and let  $\mathcal{S}$  be a set of augmented patterns such that for each  $P_R \in \mathcal{S}$  the relation  $R$  has arity  $m$ . Let  $\text{Rel}$  be a partial function that maps an instance  $I$  to a relation  $R_I$  of arity  $m$  over the points of  $\text{Patt}(I)$ .*

*We denote by  $\text{CSP}_{\overline{\text{SP}}}(\mathcal{S}, \text{Rel})$  the set of all binary CSP instances  $I$  such that  $\text{Rel}(I)$  is defined and for all  $P_R \in \mathcal{S}$  it is not the case that  $P_R \xrightarrow{SP} \text{Patt}(I)_{\text{Rel}(I)}$ .*

*We denote by  $\text{CSP}_{\overline{\text{TM}}}(\mathcal{S}, \text{Rel})$  the set of all binary CSP instances  $I$  such that  $\text{Rel}(I)$  is defined and for all  $P_R \in \mathcal{S}$  it is not the case that  $P_R \xrightarrow{TM} \text{Patt}(I)_{\text{Rel}(I)}$ .*

One of the simplest ways to augment a pattern  $P$  is by adding a binary disequality relation,  $\neq$ , to specify that some points of  $P$  are distinct. A homomorphism from an augmented pattern  $P_{\neq}$  to an augmented pattern  $Q_{\neq}$  must map points that are specified to be distinct in  $P$  to points that are specified to be distinct in  $Q$ . In the next three theorems, we shall assume that for any instance  $I$ , *all* points in  $\text{Patt}(I)_{\neq}$  are specified to be distinct. In other words, we shall assume that for any instance  $I$  the function  $\text{Rel}$  introduced in Definition 7.3 always returns the binary relation  $\neq$  containing all pairs of distinct points of  $I$ . We will denote this relation by  $\text{Rel}_{\neq}$ .

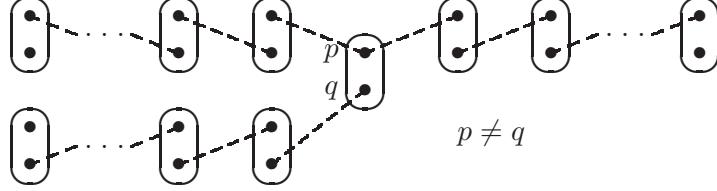


Figure 10: The augmented pattern  $\text{Pivot}_{\neq}(k)$ .

Now consider the augmented pattern  $\text{Pivot}_{\neq}(k)$  which is obtained from the pattern  $\text{Pivot}(k)$  defined in Definition 3.11 by adding a disequality relation specifying that the two points in the central node are distinct, as shown in Figure 10. Forbidding this pattern from occurring as a sub-pattern results in a larger class of instances than forbidding the pattern  $\text{Pivot}(k)$ , but our next result shows that this larger class is still tractable.

**Theorem 7.4** *The augmented pattern  $\text{Pivot}_{\neq}(k)$ , shown in Figure 10, is sub-pattern tractable.*

**Proof:** Let  $I \in \text{CSP}_{\overline{\text{SP}}}(\text{Pivot}_{\neq}(k), \text{Rel}_{\neq})$  for some constant  $k$ . If  $\text{Patt}(I)$  has a point  $x_{v,a}$  which belongs to no negative edge (i.e., it is compatible with all assignments to all other variables), then we can clearly remove all points in the same part as  $x_{v,a}$  without introducing the pattern or affecting the existence of a solution. Thus we can assume without loss of generality that  $\text{Patt}(I)$  contains no such points. A similar remark holds if  $\text{Patt}(I)$  has any parts containing just a single point.

We can also assume without loss of generality that the constraint graph of  $I$  is connected. A variable  $v$  is called an *articulation variable* of  $I$  if removing  $v$  from  $I$  disconnects the constraint graph of  $I$ . Any instance can be decomposed into a tree of components which only intersect at articulation variables. It therefore suffices to show that any instance  $I$  without articulation variables can be solved in polynomial time, so we shall assume that  $I$  has no articulation variables.

If  $\text{Pivot}(2k)$  does not occur as a sub-pattern in  $\text{Patt}(I)$  then, by Theorem 3.12 we have that  $I$  is tractable.

To deal with the remaining case, assume that  $\text{Pivot}(2k)$  occurs as a sub-pattern in  $\text{Patt}(I)$  with the central part  $U$  of  $\text{Pivot}(2k)$  mapping to part  $V$  of  $\text{Patt}(I)$ . Let  $S_{2k}$  be the set of parts of  $\text{Patt}(I)$  to which the parts of  $\text{Pivot}(2k)$  are mapped.

Since  $\text{Pivot}_{\neq}(k)$  does not occur as a sub-pattern in  $\text{Patt}(I)_{\neq}$  (and hence neither does  $\text{Pivot}_{\neq}(2k)$ ), the two points in the central part  $U$  of  $\text{Pivot}(2k)$  must map to the *same* point in  $\text{Patt}(I)$ , which we denote by  $x_{v,a}$ .

By our assumptions, we know that there is another (distinct) value  $b$  in the domain of  $v$  which belongs to a negative edge in  $\text{Patt}(I)$ , connecting part  $V$  to some other part  $W$ . If  $W$  is only connected to  $S_{2k}$  in the constraint graph of  $\text{Patt}(I)$  via  $V$ , then  $v$  is an articulation variable of  $I$ , which contradicts our assumption. Hence, there is a path  $\pi$  in the constraint graph of  $\text{Patt}(I)$  linking  $W$  to some part  $Y \in S_{2k}$  such that  $Y \neq V$ .

By choosing  $\pi$  to be minimal, we can assume that no other parts on the path  $\pi$  belong to  $S_{2k}$ . Now, since  $Y$  must lie on one of the three branches of the occurrence of  $\text{Pivot}(2k)$  in

$\text{Patt}(I)$ , we can extend  $\pi$  by following this branch from  $Y$  either towards or away from the central part  $V$ , in order to obtain a path of length at least  $k$ . This length- $k$  path, together with the first  $k$  variables of the other two branches of  $\text{Pivot}(2k)$ , gives an occurrence of the pattern  $\text{Pivot}_{\neq}(k)$  in  $\text{Patt}(I)_{\neq}$ , which contradicts our choice of  $I$ , so we are done. ■

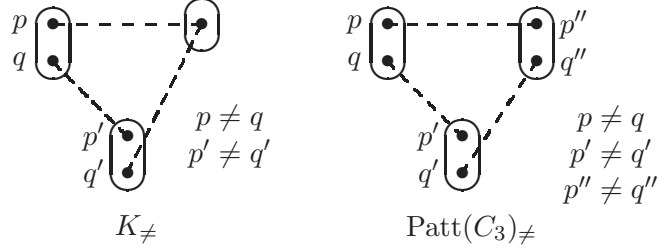


Figure 11: Two augmented patterns which are topological-minor tractable.

Now consider the augmented pattern  $K_{\neq}$ , shown in Figure 11, which is obtained from the pattern  $K$  shown in Figure 5 by adding a disequality relation to specify that any two points in the same part are distinct. We now show that forbidding  $K_{\neq}$  from occurring as a topological minor results in a tractable class (which is larger than the class obtained by forbidding the pattern  $K$  as a topological minor discussed in Theorem 5.1).

**Theorem 7.5** *The augmented pattern  $K_{\neq}$ , shown in Figure 11, is sub-pattern NP-complete but topological-minor tractable.*

**Proof:** By Theorem 3.12, the (negative) pattern  $K$  shown in Figure 5 is sub-pattern NP-complete. Since  $\text{CSP}_{\overline{\text{SP}}}(K) \subseteq \text{CSP}_{\overline{\text{SP}}}(K_{\neq}, \text{Rel}_{\neq})$ , we have that  $K_{\neq}$  is also sub-pattern NP-complete.

To show that  $K_{\neq}$  is topological-minor tractable we will show that establishing arc-consistency is sufficient to decide the existence of a solution for any instance in  $\text{CSP}_{\overline{\text{TM}}}(K_{\neq}, \text{Rel}_{\neq})$ .

By Lemma 3.7, without loss of generality we need consider only arc consistent instances. We will show, by induction on the number of variables, that in any arc-consistent instance  $I \in \text{CSP}_{\overline{\text{TM}}}(K_{\neq}, \text{Rel}_{\neq})$ , any assignment to a single variable can be extended to a solution of  $I$ . This is certainly true for instances on up to two variables, by the definition of arc consistency.

Now assume that  $I$  has more than two variables, and consider the assignment of the value  $a$  to the variable  $v$ . Let  $I[v = a]$  be the instance obtained from  $I$  by making this assignment, eliminating variable  $v$  and eliminating from the domain of all other variables  $w$  all values  $b$  such that  $(a, b) \notin R_{vw}$ . By arc consistency, none of the resulting domains in  $I[v = a]$  is empty, i.e., for each variable  $w$  there is a value  $c_w$  in the domain of  $w$  such that  $(a, c_w) \in R_{vw}$ . By the absence of  $K_{\neq}$  as a topological minor in  $\text{Patt}(I)_{\neq}$ , we can deduce that all variables  $w$  that were connected to  $v$  in the constraint graph of  $I$  are not connected in the constraint graph of  $I[v = a]$ .

Let  $S_1, \dots, S_m$  be the connected components of the constraint graph of  $I[v = a]$ . For any  $k = 1, \dots, m$ , consider the subinstance  $I[S_k]$  of the original instance  $I$  on the variables of  $S_k$ . Clearly, each  $I[S_k] \in \text{CSP}_{\overline{\text{TM}}}(K_{\neq}, \text{Rel}_{\neq})$  and each  $I[S_k]$  is arc-consistent. Furthermore, since at least the variable  $v$  has been eliminated from the original set of variables, we know that each  $I[S_k]$  has strictly fewer variables than  $I$  (even if  $m = 1$ ). Hence, by our inductive hypothesis, the assignment of any value  $c_w$  to any variable  $w$  in  $I[S_k]$  can be extended to a solution  $s_k$  to  $I[S_k]$ . The solutions  $s_k$  ( $k = 1, \dots, m$ ) together with the assignment of  $a$  to  $v$  then form a solution to  $I$  and the result follows by induction. ■

Now consider the augmented pattern  $\text{Patt}(C_3)_{\neq}$ , shown in Figure 11, which is obtained from the pattern  $\text{Patt}(C_3)$  shown in Figure 2 by adding a disequality relation specifying that



any two points in the same part are distinct. We now show that forbidding  $\text{Patt}(C_3)_\neq$  from occurring as a topological minor results in a tractable class (which is larger than the class of acyclic instances obtained by forbidding the pattern  $\text{Patt}(C_3)$  as a topological minor discussed in Proposition 4.3).

**Theorem 7.6** *The augmented pattern  $\text{Patt}(C_3)_\neq$ , shown in Figure 11, is sub-pattern NP-complete but topological-minor tractable.*

**Proof:** By Theorem 3.12, the (negative) pattern  $\text{Patt}(C_3)$  shown in Figure 2 is sub-pattern NP-complete. Since  $\text{CSP}_{\overline{\text{SP}}}(\text{Patt}(C_3)) \subseteq \text{CSP}_{\overline{\text{SP}}}(\text{Patt}(C_3)_\neq, \text{Rel}_\neq)$ , we have that  $\text{Patt}(C_3)_\neq$  is also sub-pattern NP-complete.

Singleton arc consistency (SAC) is an operation which consists in applying the following operation on an instance  $I$  until convergence: if the instance  $I[v = a]$  obtained by making the assignment of the value  $a$  to the variable  $v$  and establishing arc consistency is empty, then eliminate  $a$  from the domain of  $v$  in  $I$ . To show that  $\text{Patt}(C_3)_\neq$  is topological-minor tractable we will show that SAC is a decision procedure for  $\text{CSP}_{\overline{\text{TM}}}(\text{Patt}(C_3)_\neq, \text{Rel}_\neq)$ .

Since establishing SAC cannot introduce any occurrence of the pattern, we need only consider instances that are singleton-arc-consistent (i.e., where no more eliminations are possible by SAC). We will show, by induction on the number of variables, that in any singleton-arc-consistent instance  $I \in \text{CSP}_{\overline{\text{TM}}}(\text{Patt}(C_3)_\neq, \text{Rel}_\neq)$ , any assignment to a single variable can be extended to a solution to  $I$ . This is certainly true for instances on up to two variables, by the definition of arc consistency.

Now assume that  $I$  has more than two variables, and consider the assignment of the value  $a$  to the variable  $v$ . Let  $N$  be the set of parts of  $\text{Patt}(I)$  that are connected by a negative edge to  $x_{v,a}$ . We can assume that  $N \neq \emptyset$ , otherwise we could make the assignment  $a$  to variable  $v$  without affecting the rest of the instance  $I$ , and thus reduce  $I$  to an instance on fewer variables (which by our inductive hypothesis would have a solution).

Now let  $I[N]$  be the subinstance of  $I$  on the variables corresponding to parts in  $N$ , with the domain of each variable  $w$  of  $I[N]$  reduced to those values  $c$  such that  $(a, c) \in R_{vw}$ . Since  $I$  is singleton arc-consistent,  $I[N]$  is arc-consistent.

Let  $J'_\neq$  be the augmented pattern shown in Figure 12. Note that  $J'_\neq \xrightarrow{SP} \text{Patt}(C_3)_\neq$ . Now,

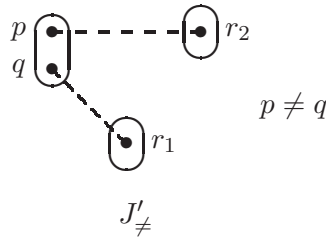


Figure 12: The augmented pattern  $J'_\neq$  used in the proof of Theorem 7.6

since  $\text{Patt}(C_3)_\neq$  does not occur as a topological minor in  $\text{Patt}(I)_\neq$ , we can deduce that  $J'_\neq$  does not occur as a topological minor in  $\text{Patt}(I[N])$ . Hence,  $K_\neq$  does not occur as a topological minor in  $\text{Patt}(I[N])$  either, since  $J'_\neq \xrightarrow{SP} K_\neq$ . By the proof of Theorem 7.5, any arc-consistent instance in  $\text{CSP}_{\overline{\text{TM}}}(K_\neq, \text{Rel}_\neq)$  has a solution, so  $I[N]$  has a solution which we denote by  $s_N$ .

Let  $u$  be a variable of  $I[N]$  and denote by  $a_u$  the value assigned to  $u$  by  $s_N$ . Let  $I_u$  be the subinstance of  $I$  on all variables of  $I$  except  $\{v\} \cup (N \setminus \{u\})$ .

Let  $S_u$  be the set of variables  $w$  of  $I_u$  which are either (1)  $u$  itself, (2) directly constrained by the assignment of  $a_u$  to  $u$  (i.e., variables  $w$  such that  $(a_u, b) \notin R_{uw}$  for some  $b$  in the domain of  $w$ ), or (3) such that the pattern  $J'_\neq$  occurs as a topological minor in  $\text{Patt}(I_u)_\neq$  with the point  $r_1$  of  $J'_\neq$  mapping to  $x_{u,a_u}$  and the point  $r_2$  of  $J'_\neq$  mapping to some point  $x_{w,b}$  for some  $b$ .

Let  $I[S_u]$  be the subinstance of  $I$  on the set of variables  $S_u$ . Clearly  $I[S_u]$  is singleton arc-consistent (since  $I$  is), and has fewer variables than  $I$  (since  $v \notin S_u$ ). Hence, by our inductive hypothesis, the assignment of value  $a_u$  to variable  $u$  can be extended to a solution  $s_u$  of  $I[S_u]$ .

Now let  $u' \in N \setminus \{u\}$ . By the absence of  $\text{Patt}(C_3)_\neq$  as a topological minor in  $\text{Patt}(I)$ , we can deduce that no assignment in  $s_u$  can be incompatible with any assignment to a variable  $y$  in  $S_{u'} \setminus S_u$ , except possibly in the case that the assignment to  $y$  is directly incompatible with both the assignment of  $a_u$  to  $u$  and  $a_{u'}$  to  $u'$ . In this latter case, the solution  $s_{u'}$  projected onto  $S_{u'} \setminus S_u$  is necessarily consistent with  $s_u$ .

Hence, by a simple inductive argument, we can create a consistent partial assignment composed of the assignment of  $a$  to  $v$ , and the assignments specified by  $s_N$  and each  $s_u$  (projected onto the not-yet-assigned variables).

The rest of the instance  $I$ , if it is non-empty, is not constrained by this partial assignment and by our inductive hypothesis has a solution; combining these partial solutions gives a solution to  $I$ . ■

Classes of the CSP that are defined by specifying a restricted set of constraint relations over some fixed domain  $D$  are known as *language classes* [29, 22]. Every known tractable language class [29, 2] of CSP instances is characterised by an operation  $f : D^k \rightarrow D$  with the property that for all constraints  $R_{uv}$ , and all pairs  $(p_1, q_1), (p_2, q_2), \dots, (p_k, q_k) \in R_{uv}$ , the pair  $(f(p_1, p_2, \dots, p_k), f(q_1, q_2, \dots, q_k)) \in R_{uv}$ ; such an operation is known as a *polymorphism* of the constraint relations [2, 29].

We now show that using augmented patterns we can characterise every known tractable language class using a single forbidden augmented sub-pattern.

**Theorem 7.7** *Every tractable language class of binary CSP instances that is characterised by a polymorphism  $f$  is equal to  $\text{CSP}_{\overline{\text{SP}}}(P_R, \text{Rel}_f)$  for some augmented pattern  $P_R$  and function  $\text{Rel}_f$ .*

**Proof:** The  $k$ -ary operation  $f : D^k \rightarrow D$  can be specified by a  $(k+1)$ -ary relation  $R_f$  over  $D$  where  $R_f = \{(a_1, \dots, a_{k+1}) \mid a_{k+1} = f(a_1, \dots, a_k)\}$ . Define  $\text{Rel}_f$  to be the function that maps any CSP instance  $I$  over  $D$  to the relation  $R$  over the points of  $\text{Patt}(I)$ , where  $R = \{(x_{v,a_1}, \dots, x_{v,a_{k+1}}) \mid (a_1, \dots, a_{k+1}) \in R_f\}$ .

The class of all instances  $I$  over domain  $D$  for which all constraint relations admit  $f$  as a polymorphism, is precisely the class of instances defined by  $\text{CSP}_{\overline{\text{SP}}}(P_R, \text{Rel}_f)$  where  $P = (X, E^\sim, E^+, E^-)$  with

- $X = U \cup V$ , where  $U = \{p_1, p_2, \dots, p_{k+1}\}$  and  $V = \{q_1, q_2, \dots, q_{k+1}\}$ ;
- $E^\sim = (U \times U) \cup (V \times V)$ ;
- $E^+ = \{(p_i, q_i) \mid p_i \in U, q_i \in V, i = 1, 2, \dots, k\}$ ;
- $E^- = \{(p_{k+1}, q_{k+1})\}$ ;

and  $R = \{(p_1, p_2, \dots, p_{k+1}), (q_1, q_2, \dots, q_{k+1})\}$ , as illustrated in Figure 13. ■

We remark that the algebraic dichotomy conjecture [5], which is a refinement of the dichotomy conjecture of Feder and Vardi [22], implies that *every* tractable language is characterised by a single polymorphism, and thus under this conjecture Theorem 7.7 applies to *all* tractable language classes of binary CSP instance over a fixed domain.

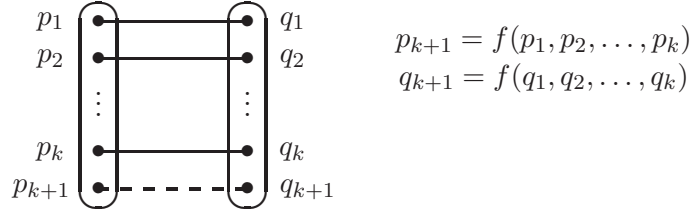


Figure 13: The augmented pattern  $P_R$  used in the proof of Theorem 7.7.

## 8 Conclusions and open problems

The notion of a pattern occurring as a topological minor, introduced here, allows a new approach to the definition of tractable classes of CSP instances. We have shown that this approach, together with the notion of augmented patterns, can unify the description of all tractable structural and language classes, as well as allowing new and more general tractable classes to be identified. We therefore believe that it has great potential for systematically identifying all tractable classes of the CSP.

One long-term goal is to characterise precisely which patterns  $P$  are topological-minor tractable and for which such patterns  $P$ ,  $\text{CSP}_{\text{TM}}(P)$  is recognisable in polynomial time. For example, Figure 14 shows three simple patterns whose topological minor tractability is currently open.

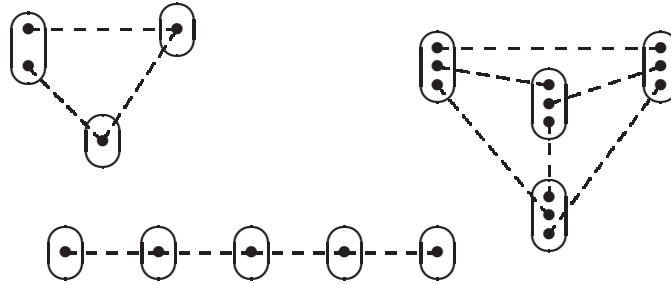


Figure 14: Three patterns whose topological-minor tractability is open.

Another avenue of future research is the discovery of other applications for topological minors, such as in variable elimination [9]. Indeed, perhaps the most interesting open question is whether the notion of topological minor, introduced in this paper, will find applications other than the definition of tractable classes of the CSP. We have seen that certain classic results from graph theory can lead to results concerning topological minors of CSP instances. An intriguing avenue for future research is to build bridges in the other direction. For example, a corollary of the proof of Theorem 6.1 is that finding a path linking two given vertices and which passes at most once through each part of an  $n$ -partite graph is NP-hard. Another way of expressing this is that finding a *heterochromatic path* linking two given vertices in a vertex-coloured graph is NP-hard [32, 4].

To achieve further progress it may well be necessary to further refine or modify the definition of a topological minor given here. We regard this work as simply a first step towards a general topological theory of complexity for constraint satisfaction problems.

## References

- [1] Stefan Arnborg, Andrzej Proskurowski, and Derek G. Corneil. Forbidden minors characterization of partial 3-trees. *Discrete Mathematics*, 80(1):1–19, 1990.

- [2] Libor Barto. Constraint satisfaction problem and universal algebra. *ACM SIGLOG News*, 1(2):14–24, 2014.
- [3] Libor Barto and Marcin Kozik. Constraint satisfaction problems solvable by local consistency methods. *J. ACM*, 61(1):3, 2014.
- [4] Hajo Broersma, Xueliang Li, Gerhard Woeginger, and Shenggui Zhang. Paths and cycles in colored graphs. *Australasian Journal of Combinatorics*, 31:299–311, 2005.
- [5] Andrei Bulatov, Andrei Krokhin, and Peter Jeavons. Classifying the complexity of constraints using finite algebras. *SIAM Journal on Computing*, 34(3):720–742, 2005.
- [6] Clément Carbonnel and Martin C. Cooper. Tractability in constraint satisfaction problems: a survey. *Constraints*, 21(2):115–144, 2016.
- [7] D. Cohen. A new hybrid class for which arc-consistency is a decision procedure. In *Proceedings of 9th International Conference on Principles and Practice of Constraint Programming (CP’03)*, volume 2833 of *Lecture Notes in Computer Science*, pages 807–811. Springer-Verlag, September 2003.
- [8] David A. Cohen, Martin C. Cooper, Páidí Creed, Dániel Marx, and András Z. Salamon. The tractability of CSP classes defined by forbidden patterns. *J. Artif. Intell. Res. (JAIR)*, 45:47–78, 2012.
- [9] David A. Cohen, Martin C. Cooper, Guillaume Escamoché, and Stanislav Živný. Variable and value elimination in binary constraint satisfaction via forbidden patterns. *Journal of Computer and System Sciences*, 81(7):1127–1143, 2015.
- [10] David A. Cohen, Martin C. Cooper, Peter Jeavons, and Stanislav Živný. Tractable classes of binary CSPs defined by excluded topological minors. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI’15)*, pages 1945–1951. AAAI Press, 2015.
- [11] David A. Cohen and Peter G. Jeavons. The power of propagation: When GAC is enough. *Constraints*, 2016. To appear.
- [12] Martin C. Cooper. An optimal k-consistency algorithm. *Artif. Intell.*, 41:89–95, 1989.
- [13] Martin C. Cooper and Guillaume Escamoché. Characterising the complexity of constraint satisfaction problems defined by 2-constraint forbidden patterns. *Discrete Applied Mathematics*, 184:89–113, 2015.
- [14] Martin C. Cooper, Peter G. Jeavons, and András Z. Salamon. Generalizing constraint satisfaction on trees: Hybrid tractability and variable elimination. *Artif. Intell.*, 174(9-10):570–584, 2010.
- [15] Martin C. Cooper and Stanislav Živný. The power of arc consistency for CSPs defined by partially-ordered forbidden patterns. *Proceedings of LICS*, 2016.
- [16] R. Dechter and J. Pearl. Tree clustering for constraint networks. *Artif. Intell.*, 38:353–366, 1989.
- [17] Rina Dechter. *Constraint Processing*. Morgan Kaufmann, 2003.
- [18] Reinhard Diestel. *Graph Theory*. Springer, fourth edition edition, 2010.
- [19] G. A. Dirac. Short proof of menger’s graph theorem. *Mathematika*, 13(1):42–44, 1966.

- [20] R.G. Downey and M.R. Fellows. *Parametrized Complexity*. Springer, 1999.
- [21] G. Escamocher. *Forbidden Patterns in Constraint Satisfaction Problems*. Ph.D thesis, Toulouse University, IRIT, University of Toulouse, France, 2013.
- [22] Tomás Feder and Moshe Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through Datalog and group theory. *SIAM Journal of Computing*, 28(1):57–104, 1998.
- [23] J. Flum and M. Grohe. *Parametrized Complexity Theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2006.
- [24] E.C. Freuder. A sufficient condition for backtrack-free search. *J. ACM*, 29(1):24–32, 1982.
- [25] E.C. Freuder. A sufficient condition for backtrack-bounded search. *J. ACM*, 32(4):755–761, 1985.
- [26] Martin Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *J. ACM*, 54(1):1–24, March 2007.
- [27] Martin Grohe, Ken-ichi Kawarabayashi, Dániel Marx, and Paul Wollan. Finding topological subgraphs is fixed-parameter tractable. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA*, pages 479–488. ACM, 2011.
- [28] Pavol Hell and Jaroslav Nešetřil. Colouring, constraint satisfaction, and complexity. *Computer Science Review*, 2(3):143–163, 2008.
- [29] Peter Jeavons, David A. Cohen, and Marc Gyssens. Closure properties of constraints. *J. ACM*, 44(4):527–548, 1997.
- [30] Philippe Jégou. Decomposition of domains based on the micro-structure of finite constraint-satisfaction problems. In *AAAI*, pages 731–736, Menlo Park, CA, USA, jul 1993. AAAI Press.
- [31] Gábor Kun and Jaroslav Nešetřil. Forbidden lifts (NP and CSP for combinatorialists). *European Journal of Combinatorics*, 29(4):930–945, 2008.
- [32] Xueliang Li, Shenggui Zhang, and Hajo Broersma. Paths and cycles in colored graphs. *Electronic Notes in Discrete Mathematics*, 8:128–132, 2001.
- [33] Florent R. Madelaine and Iain A. Stewart. Constraint satisfaction, logic and forbidden patterns. *SIAM Journal on Computing*, 37(1):132–163, 2007.
- [34] Dániel Marx. Tractable hypergraph properties for constraint satisfaction and conjunctive queries. *J. ACM*, 60(6):42, 2013.
- [35] Neil Robertson and Paul D. Seymour. Graph minors. V. Excluding a planar graph. *J. Comb. Theory, Ser. B*, 41(1):92–114, 1986.
- [36] Neil Robertson and Paul D. Seymour. Graph minors. XX. Wagner’s conjecture. *J. Comb. Theory, Ser. B*, 92(2):325–357, 2004.
- [37] F. Rossi, V. Dahr, and C. Petrie. On the equivalence of constraint satisfaction problems. In *Proceedings of the European Conference on Artificial Intelligence (ECAI90)*, August 1990. Also MCC Technical Report ACT-AI-222-89.

- [38] Francesca Rossi, Peter van Beek, and Toby Walsh, editors. *The Handbook of Constraint Programming*. Elsevier, 2006.
- [39] William T. Tutte. *Connectivity in Graphs*. University of Toronto Press, 1966.